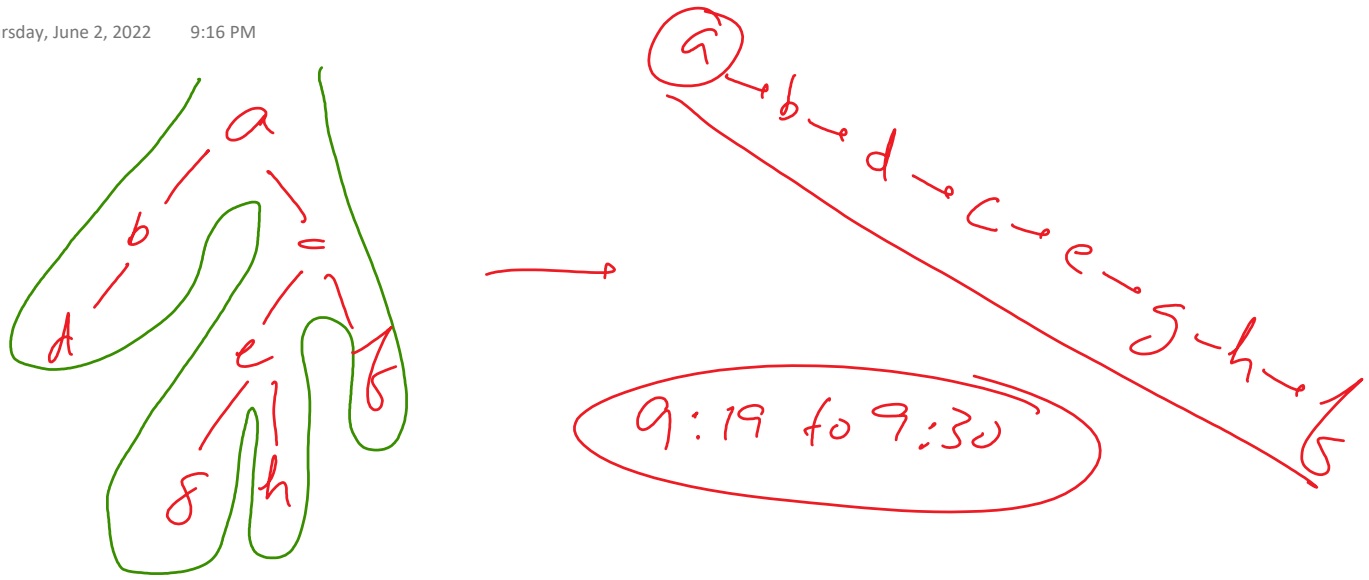
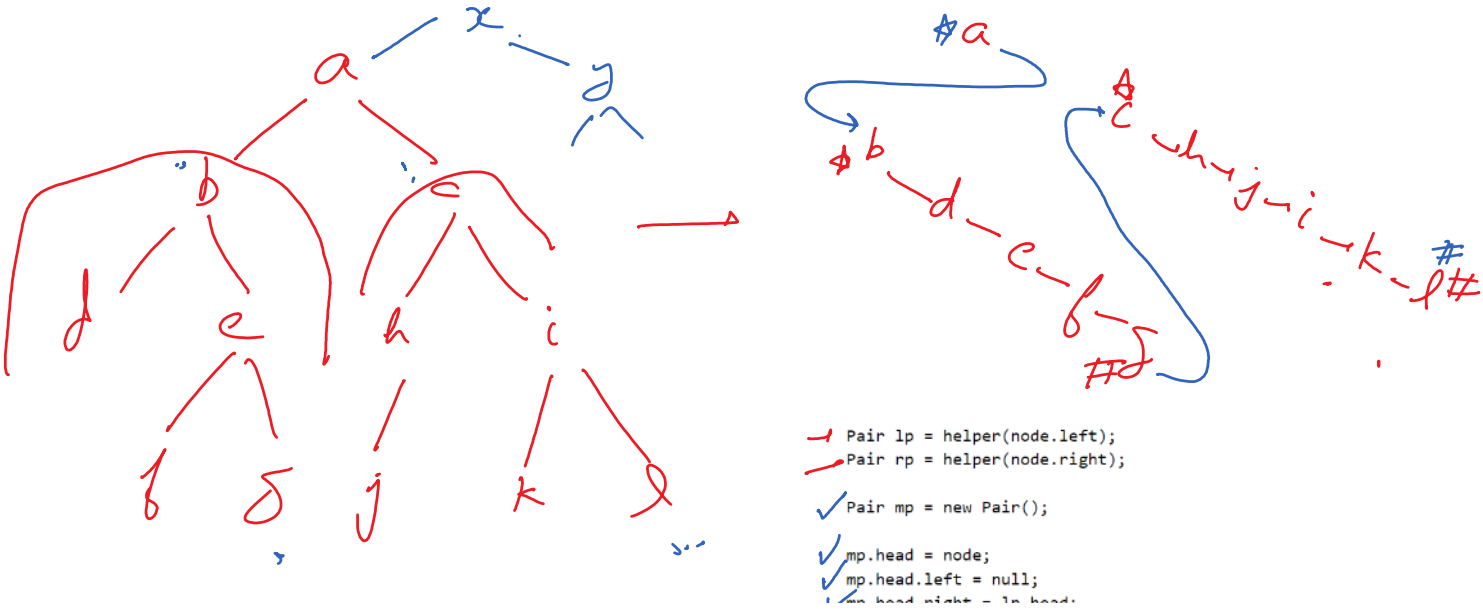
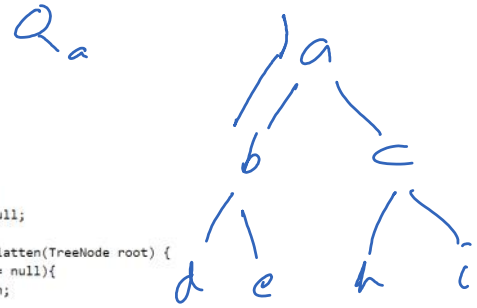
Thursday, June 2, 2022 9:16 PM

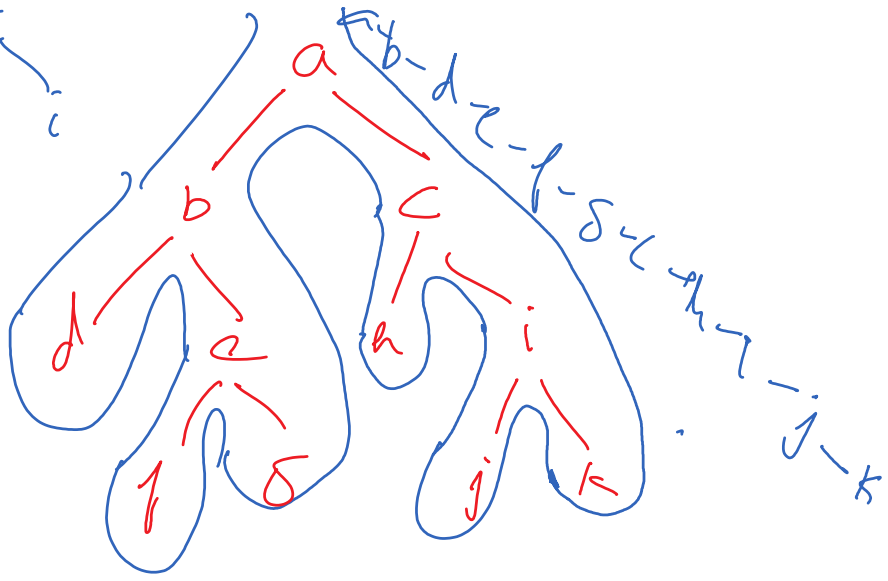


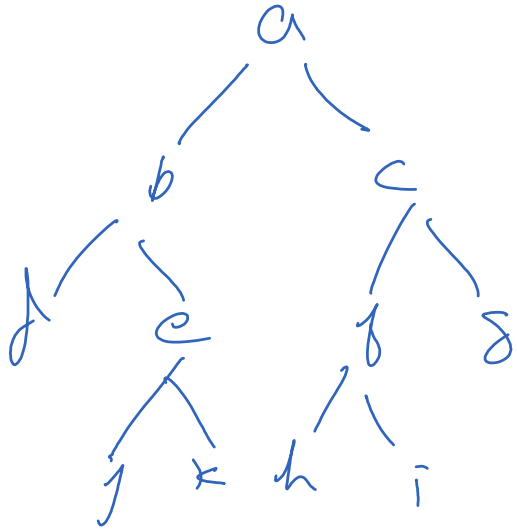


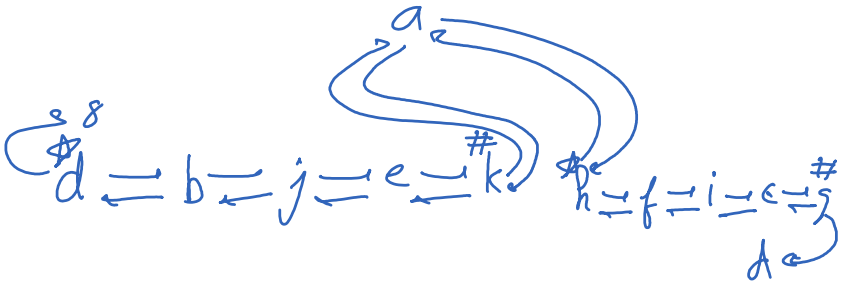
```
Pair mp = new Pair();

mp.head = node;
mp.head.left = null;
mp.head.right = lp.head;
lp.tail.right = rp.head;
mp.tail = rp.tail;

return mp;
```



```
class Solution {
    Node prev = null;

    public void flatten(TreeNode root) {
        if(root == null){
            return;
        }

        flatten(root.right);
        flatten(root.left);

        root.left = null;
        root.right = prev;
        prev = root;
    }
}
```

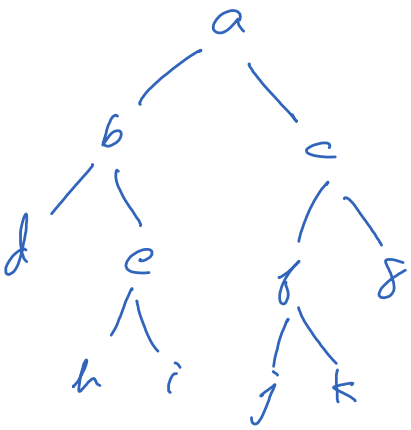


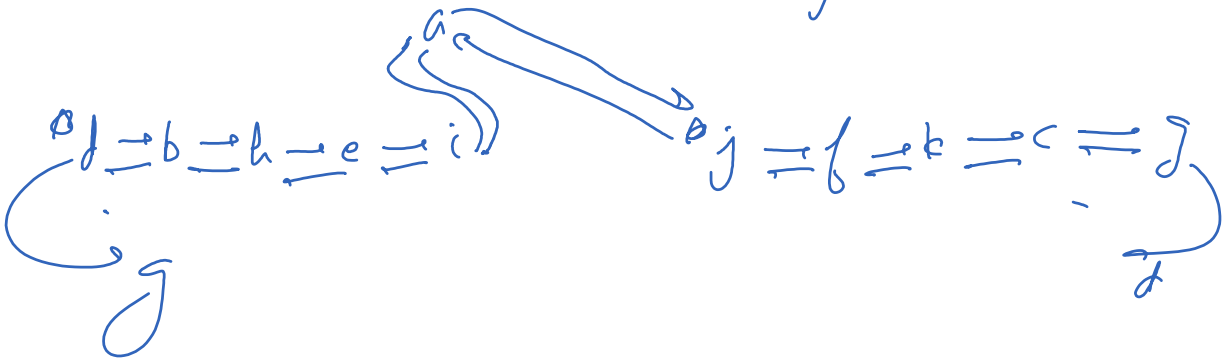




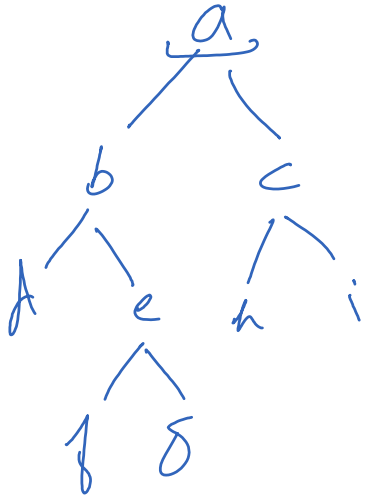
```
Node helper(Node node){
    if(node == null){
        return null;
    }

    Node lhead = helper(node.left)
    Node rhead = helper(node.right

    Node onl = node;
    onl.left = onl.right = onl;

    Node s1 = concat(lhead, onl);
    Node s2 = concat(s1, rhead);

    return s2;
}
```

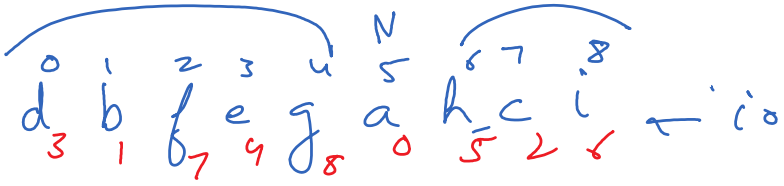
```
Node concat(Node h1, Node h2){
    if(h1 == null){
        return h2;
    } else if(h2 == null){
        return h1;
    }

    Node t1 = h1.left;
    Node t2 = h2.left;

    t1.right = h2;
    h2.left = t1;

    t2.right = h1;
    h1.left = t2;

    return h1;
}
```

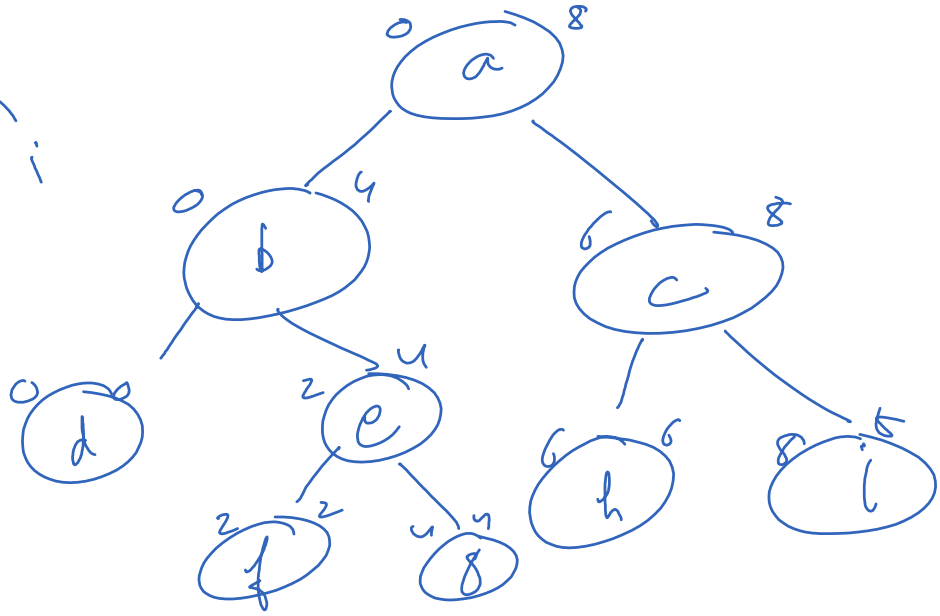




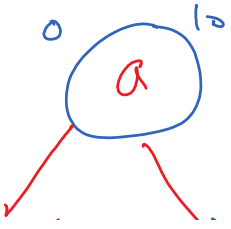
a - 0
b - 1
c - 2
d - 3
e - 4
h - 5
i - 6
f - 7
g - 8

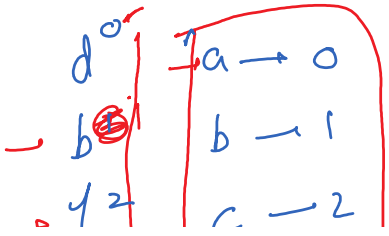




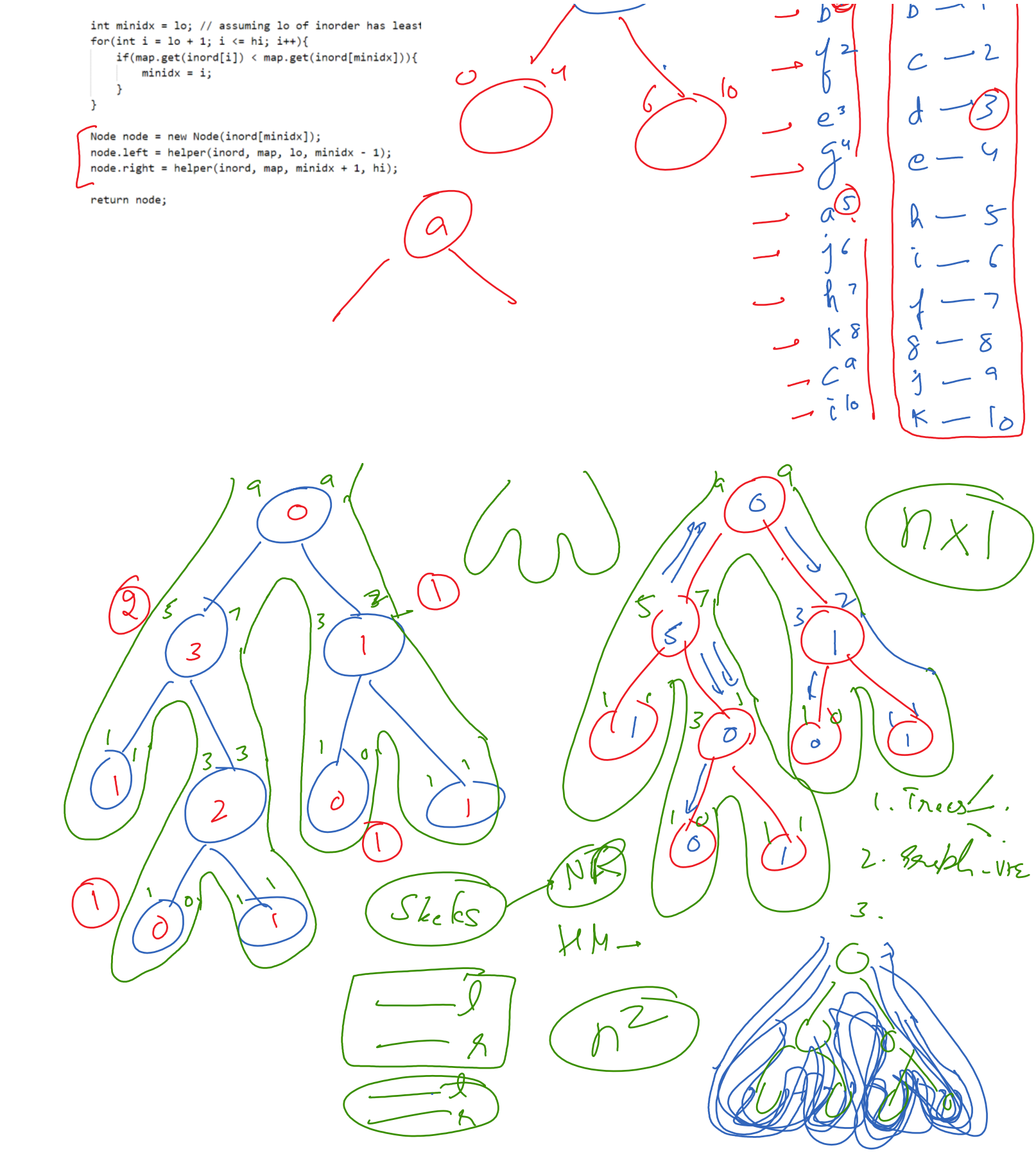


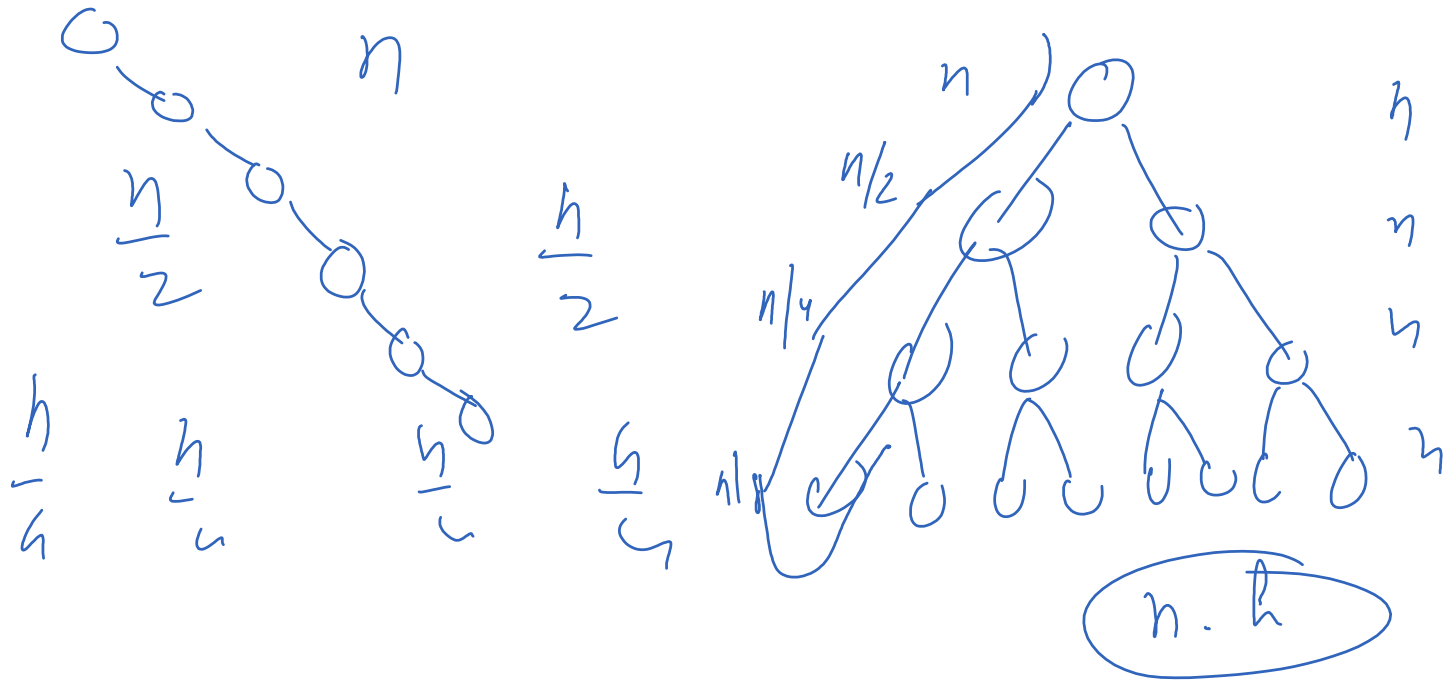

```
if(lo > hi){
    return null;
}

int minidx = lo; // assuming lo of inorder has least
for(int i = lo + 1; i <= hi; i++){
    if(map.get(inord[i]) < map.get(inord[minidx])){
```

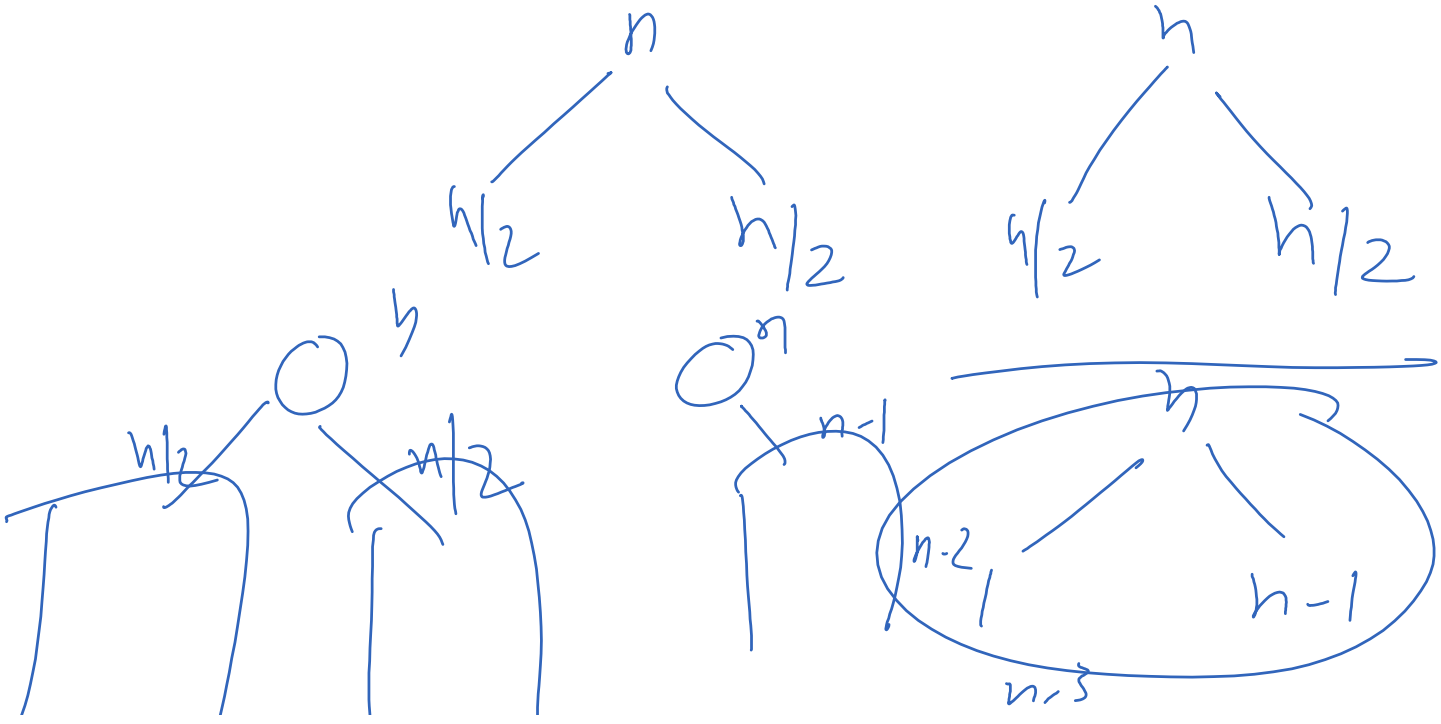




```
int minidx = lo; // assuming lo of inorder has least
for(int i = lo + 1; i <= hi; i++){
    if(map.get(inord[i]) < map.get(inord[minidx])){
        minidx = i;
    }
}

Node node = new Node(inord[minidx]);
node.left = helper(inord, map, lo, minidx - 1);
node.right = helper(inord, map, minidx + 1, hi);

return node;
```




n
n/2
n/4
n
n
n
n
n/2
n/2
n/2
n/2
n/4
n . h


n
n/2
n/2
n
n/2
n/2
n/2
n/2
n
n-1
n
n-2
n-1
n-3

$n-3$

$$T(n) = T(n_2) + T(n_1)$$